图文 078、动手实验：自己模拟出JVM堆内存溢出的场景体验一下！

详情 评论

动手实验：
自己模拟出JVM堆内存溢出的场景体验一下！

狸猫技术窝专栏上新，基于**真实订单系统**的消息中间件（mq）实战，重磅推荐：

未来3个月，我的好朋友原子弹大侠将带你一起，全程实战，360度死磕MQ

（点击下方蓝字进行试听）

从 0 开始带你成为消息中间件实战高手

重要说明：

如何提问：每篇文章都有评论区，大家可以尽情在评论区留言提问，我都会逐一答疑

（**ps**：评论区还精选了一些小伙伴对**专栏每日思考题的作答**，有的答案真的非常好！大家可以通过看别人的思路，启发一下自己，从而加深理解）

如何加群：购买了狸猫技术窝专栏的小伙伴都可以加入**狸猫技术交流群**。

（群里有不少**一二线互联网大厂的助教**，大家可以一起讨论交流各种技术）

具体**加群方式**请参见文末。

（**注**：以前通过其他专栏加过群的同学就不要重复加了）

1、前文回顾

上篇文章我们已经分析过了栈内存溢出的场景，同时用示例代码带着大家做了实验，体验了一下栈内存溢出的时候是什么样子的。

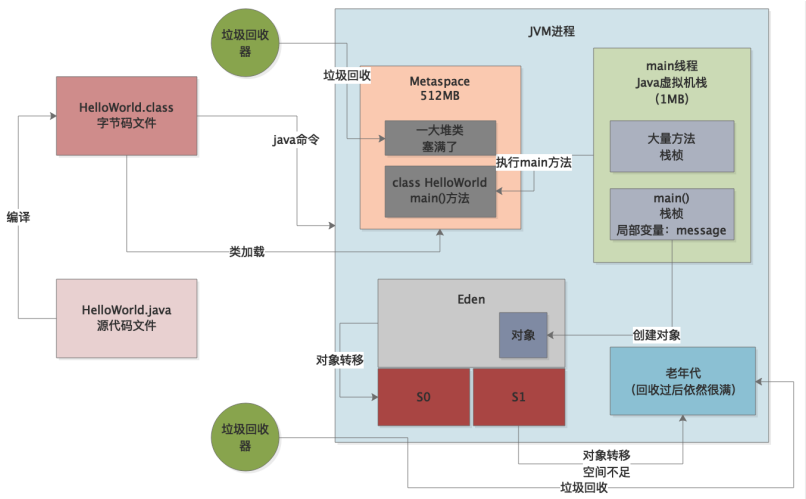
不过说实话，Metaspace区域和栈内存的溢出，一般都是极个别情况下才会发生的，并不是一种普遍的现象。

但是今天要给大家讲的这个堆内存溢出的场景，就是非常普遍的现象了，一旦要是系统负载过高，比如并发量过大，或者是数据量过大，或者是出现了内存泄漏的情况，很容易就导致JVM内存不够用了，就会堆内存溢出，然后系统崩溃。

所以今天就带着大家来体验一下堆内存溢出的场景。

2、回顾一下堆内存溢出的一个典型场景

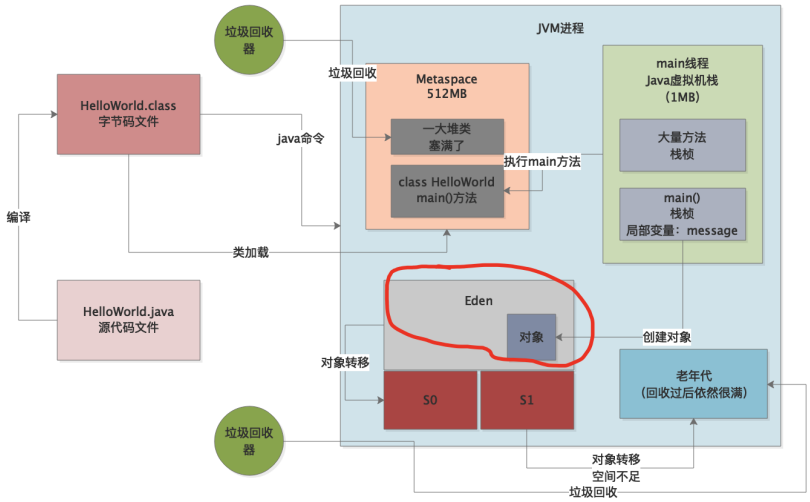
首先还是来一张完整的JVM运行原理图，里面包含了对象的分配，GC的触发，对象的转移，各个环节如何触发内存溢出的，大家一定要牢记这张图。



接着我们就来回顾一下一个典型的堆内存溢出的场景：

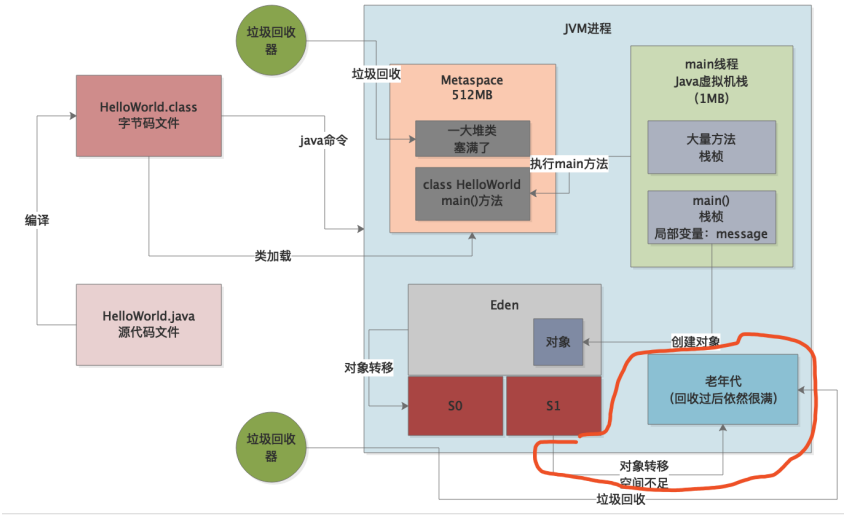
假设现在系统负载很高，不停的运转和工作，不停的创建对象塞入内存里，刚开始是塞入哪里的？

当然是年轻代的Eden区了，如下图红圈所示。



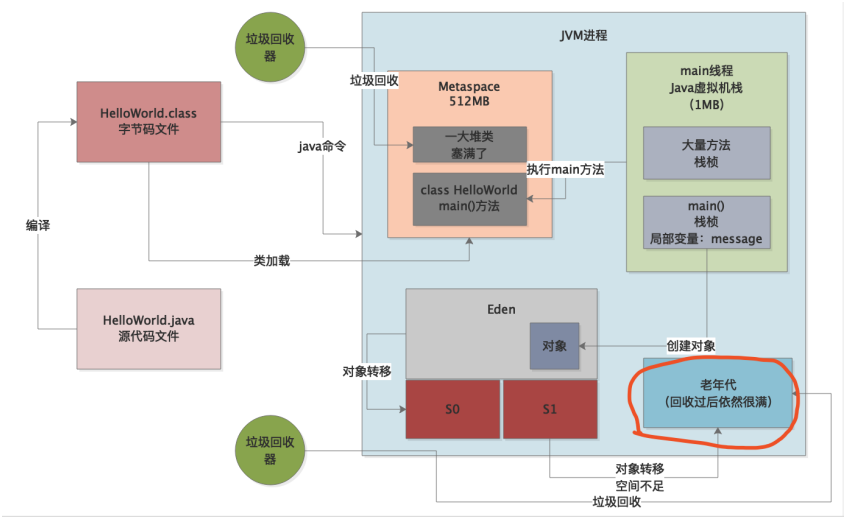
但是因为系统负载实在太高了，很快就把Eden区塞满了，这个时候触发ygc

但是ygc的时候发现不对劲，因为似乎Eden区里还有很多的对象都是存活的，而且survivor区域根本放不下，这个时候只能把存活下来的大批对象放入老年代中去，如下图红圈处。

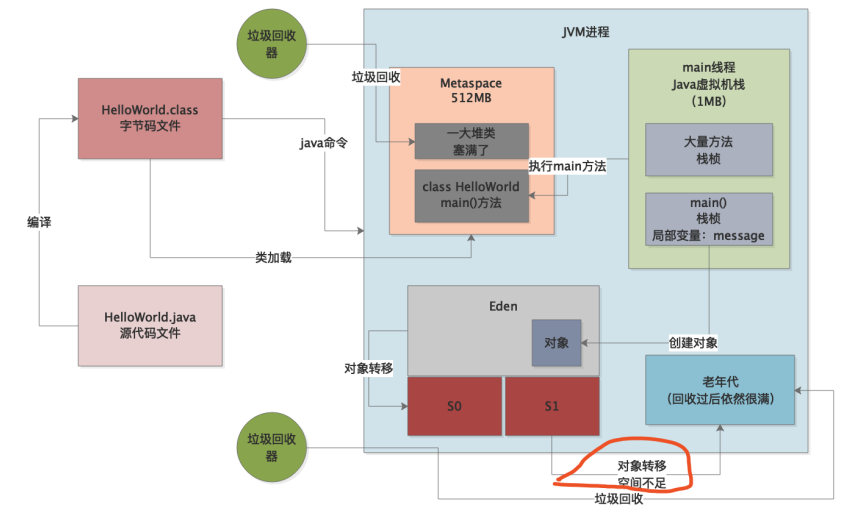


就这么来几次ygc之后，每次ygc后都有大批对象进入老年代，老年代很快就会塞满了，而且最重要的是这里的对象还大多都是存活的。

所以接下来一次ygc后又要转移一大批对象进入老年代，先触发full gc，但是full gc之后老年代里还是塞满了对象，如下图红圈所示。



这个时候ygc后存活下来的对象哪怕在full gc之后还是无法放入老年代中，此时就直接报出堆内存溢出了，如下图红圈所示。



所以堆内存溢出的场景就是这样子，非常的简单，后续我们会用真实的案例结合一些业务系统的实际情况给大家继续展示堆内存溢出的一些场景。

3、用示例代码来演示堆内存溢出的场景

我们来看下面这段代码：

```
public class Demo3 {
    public static void main(String[] args) {
        long counter = 0;
        List<Object> list = new ArrayList<Object>();
        while(true) {
            list.add(new Object());
            System.out.println("当前创建了第" + (++counter) + "个对象");
        }
    }
}
```

代码很简单，就是在一个while循环里不停的创建对象，而且对象全部都是放在List里面被引用的，也就是不能被回收。

大家试想一下，如果你不停的创建对象，Eden区满了，他们全部存活会全部转移到老年代，反复几次之后老年代满了。

然后Eden区再次满了，ygc后存活对象再次进入老年代，此时老年代先full gc，但是回收不了任何对象，因此ygc后的存活对象就一定是无法进入老年代的。

所以我们用下面的JVM参数来运行一下代码：-Xms10m -Xmx10m，我们限制了堆内存大小总共就只有10m，这样可以尽快触发堆内存的溢出。

我们在控制台打印的信息中可以看到如下的信息：

当前创建了第360145个对象
Exception in thread "main" java.lang.OutOfMemoryError: Java heap space

所以从这里就可以看到，在10M的堆内存中，用最简单的Object对象搞到老年代被塞满大概需要36万个对象。然后堆内存实在放不下任何其他对象，此时就会OutOfMemory了，而且告诉了你是Java heap space，也就是堆空间发生了内存溢出的。

4、本文总结

从这篇文章我们可以看到堆内存是如何溢出的，以及溢出的时候回看到什么样的异常信息

接下来我们将会用两天的时间给大家讲几个真实的线上生产案例，让大家感受一下我们曾经线上系统是如何遇到内存溢出的。

End
